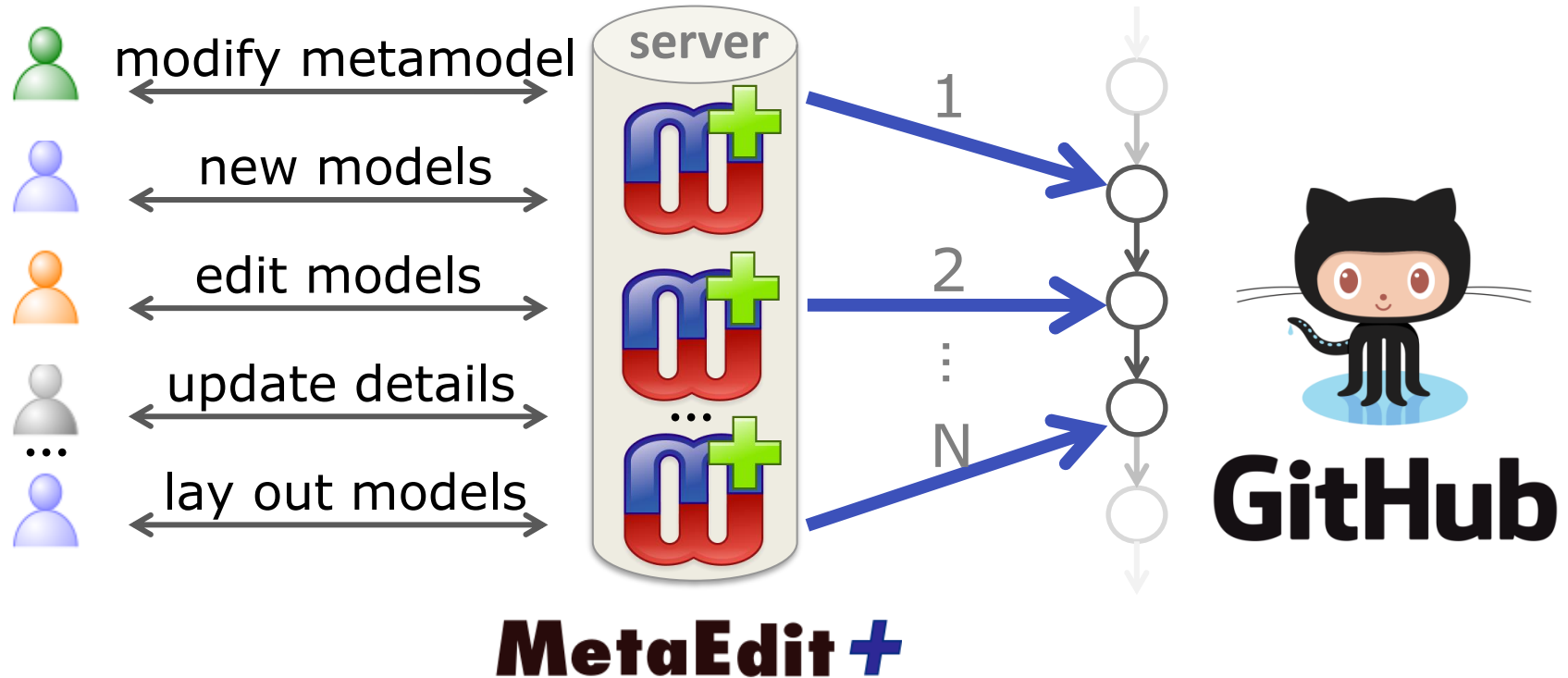


Collaborative Modeling and Metamodeling in the Cloud with Versioning

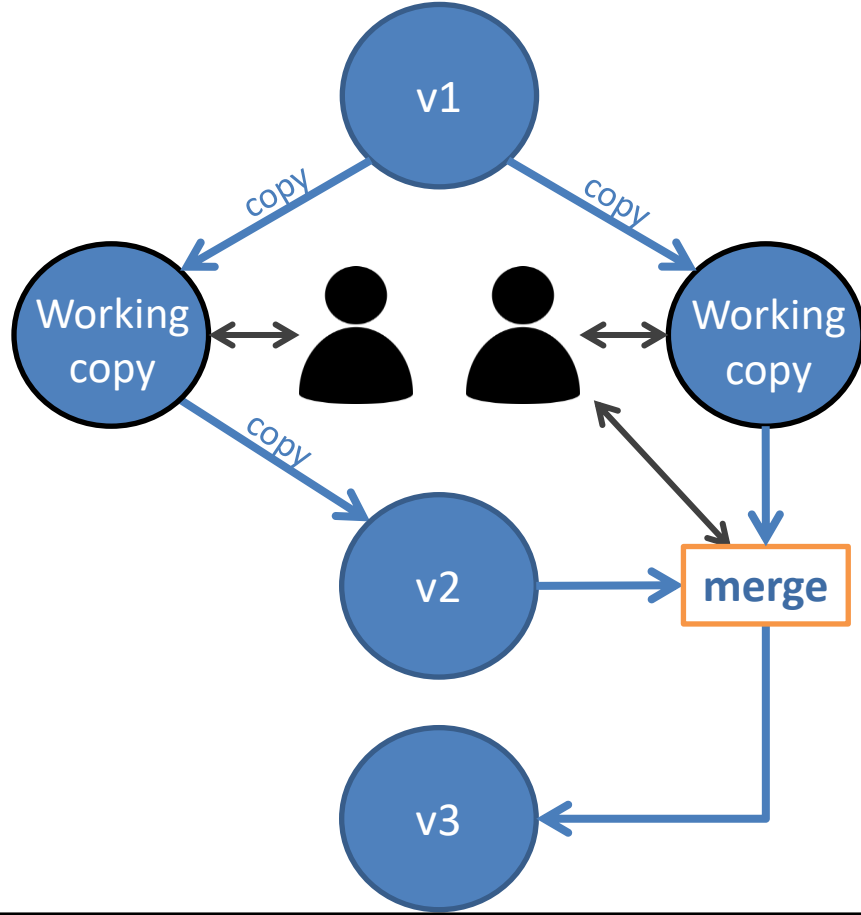
Steven Kelly, Juha-Pekka Tolvanen
stevek | jpt @metacase.com

Collaborative (meta-)Modeling and Versioning of Models and Languages



Clone

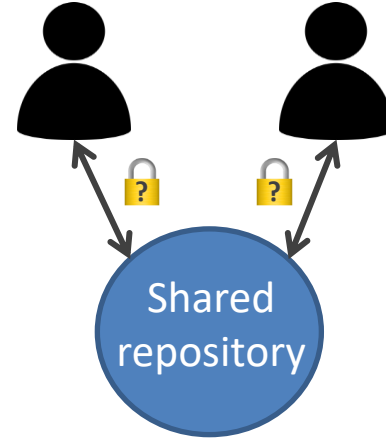
merge after the fact



vs.

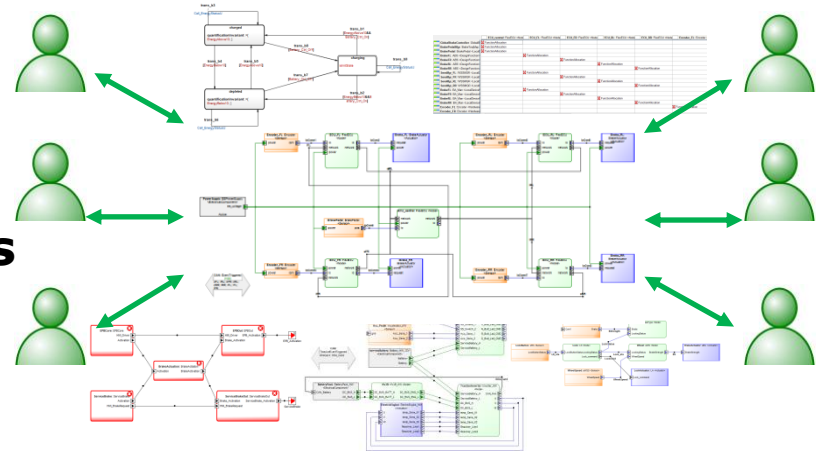
Share

continuous integration



MetaEdit+

- Mature, commercial, supported Language Workbench
- Collaborative modeling support
 - Multiple developers
 - Multiple models
 - Multiple languages
 - **Tracks history & changes**
 - **Integrated with VCSs**
- Collaborative language engineering
 - **Updates existing models**
 - **Integrates with VCSs**



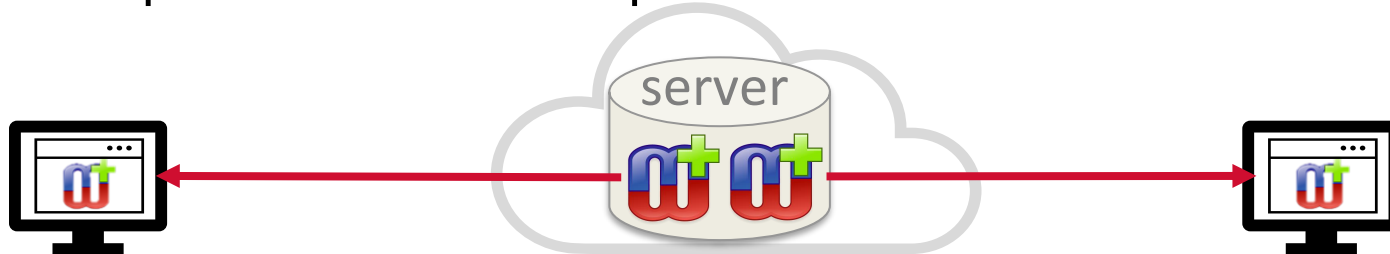
Collaboration at the object level



- Repository-based: users work in parallel on same data
 - No need to merge
 - Models and metamodels can be edited simultaneously
- Locks ensure no conflicts
 - Fine granularity, so can work closely without lockout
- Low ceremony: locking and unlocking is automatic
 - Users can focus on work, not tool
- Design transactions: minutes to hours
 - Commit a coherent set of changes together
 - Releasing half-finished/inconsistent data would confuse others
 - See a consistent version of repository during transaction
 - Avoid being distracted by others' changes appearing live

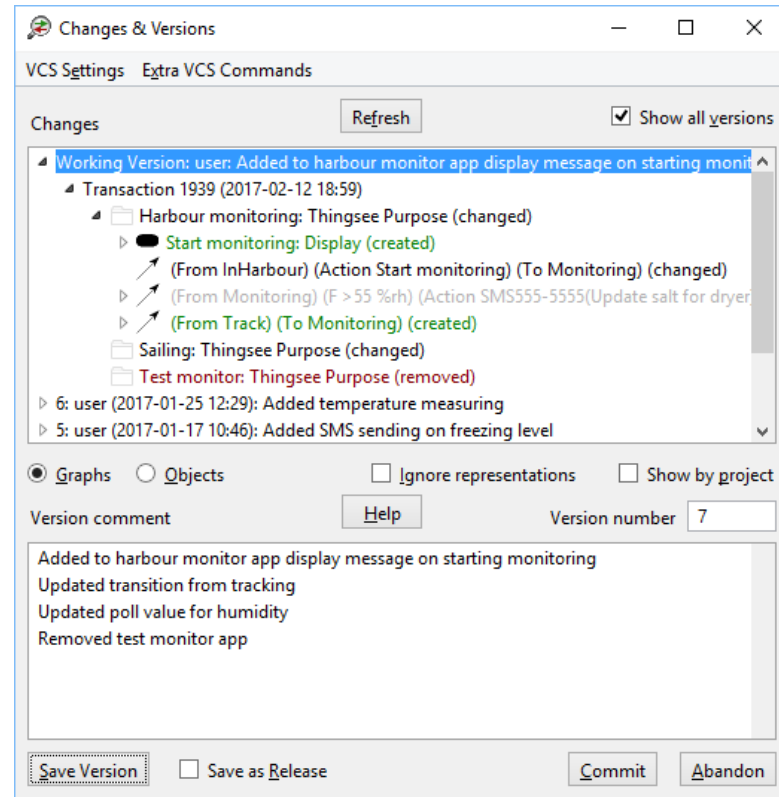
RSL: Remote Shared License

- IT departments want to avoid installing software
- Want to allow users to work from anywhere
- Graphical modelling has high performance reqs at client
- Large repositories are slow to move over a network
 - Particularly with high latency over VPNs + load as needed
- Put MetaEdit+ server and client on one Windows Server
- Multiple remote desktop connections from users



Changes and Versions tools

- Automatic trace of model changes
- See changes graphically directly in your models
- View changes as a tree using your language's structure and symbols, not XML
- Compare changes as a textual diff with live links to models
- Filters (your changes/all changes, data/representation)
- All the info you need, clearly: => easy to write version comments



Compare changes in different ways

Changes & Versions

VCS Settings Extra VCS Commands

Changes Refresh Show all versions

Working Version: user

- Transaction 383 (2017-01-25 15:05)
 - My apps: Thingsee Profile
 - Harbour monitoring: Thingsee purpose (changed)
 - Start monitoring::: Display (created)
 - 2 lux: Luminance (changed)
 - InitializeSailing: Other Purpose (changed)
 - < 0 C: Temperature
 - (From InHarbour) (Action Start monitoring!) (To Monitoring) (changed)
 - Speeding Alert: Thingsee purpose (removed)

Graphs Objects Ignore representations Show by project

Version comment Help Version number

Comparing Harbour monitoring: Thingsee purpose changes in Transaction 383 (2017-01-25 15:05)

Start state: F	Always active: F
Always active: F	Description:
Description:	Object: Monitoring <State> 3_8332
Object: < 2 C <Temperature> 3_8632	Name: Monitoring
Trigger if over (celcius):	Start state: F
Trigger if under (celcius): 2	Always active: F
Relationship: <Transition> 3_8179	Description:
Name:	Object: < 0 C <Temperature> 3_8632
Role: From <From> Object: <State> 3_8053	Trigger if over (celcius):
Role: To <To> Object: <State> 3_8332	Trigger if under (celcius): 0
Relationship: Battery warning <Transition> 3_8384	Relationship: <Transition> 3_8179
Name: Battery warning	Name:
Role: Action <Action> Object: <Action> 3_8528	Role: Action <Action> Object: <Display> 3_12278
Role: From <From> Object: <State> 3_8332	Role: From <From> Object: <State> 3_8053
Role: F <PollTrigger> Object: <Battery> 3_8594	Role: To <To> Object: <State> 3_8332

Save Version

Grid: 10 @ 10 Snap Show 100%

Versioning with external VCSs

- Simple setup and use
- Automated background execution of versioning commands
- Maintain same model in several places, syncing with GitHub etc.
- Supports multiple simultaneous versioners ...unlike some VCSs...
- Version and diff your language and generators along with models
- Git, SVN predefined
 - extend with your own

The image shows a screenshot of a web browser displaying a GitHub repository for 'mccjpt/IoT'. The repository page shows the 'master' branch with 1 branch and 0 tags. Below the repository name, there is a list of files and folders with their respective authors and dates:

File/Folder	Author	Date
metamodel	13: user	(2023-06-01 10:00Z)
versionedDB	13: user	(2023-06-01 10:00Z)
.gitattributes	Imported new e	
Boat AllSeason Monitor_3...	1: user	(2017-01-25 15:34Z)
Boat Monitor_3_7989_Thi...	1: user	(2017-01-25 15:34Z)
Bump monitoring_3_8966...	1: user	(2017-01-25 15:34Z)
Harbour monitoring_3_81...	5: user	(2017-01-30 12:46Z)

Overlaid on the right side of the browser is a window titled 'Changes & Versions'. This window shows a tree view of changes and a version comment field. The version comment field contains the following text:

```
Added to harbour monitor app display message on starting
Updated transition from tracking
Updated poll value for humidity
Removed test monitor app
```

Below the version comment field, there are two buttons: 'Save Version' and 'Save as Release'. The 'Save Version' button is highlighted with a red box.

Task overview by role (in parallel)

Metamodeler	Layout Expert	Climate Specialist	Technical Modeler	Model Creator
Add new property slot	Rearrange diagrams: - Sensors left - Actions right	Update temperatures - warmer, stabler Update velocities - rougher seas	<i>Wait for metamodeler</i>	Create new models: - storm season - tourist season
Add/update rules Update symbols			Enter values for new property	

**We'll have one or more groups (Models1, Model2...)
and several roles in each group (meta**1**, layout**1** in Models**1**...)**

Quick demo...

Task overview by role (in parallel)

Metamodeler	Layout Expert	Climate Specialist	Technical Modeler	Model Creator
Add new property slot	Rearrange diagrams: - Sensors left - Actions right	Update temperatures - warmer, stabler Update velocities - rougher seas	<i>Wait for metamodeler</i>	Create new models: - storm season - tourist season
Add/update rules Update symbols			Enter values for new property	

We'll have one or more groups (Models1, Model2...)
and several roles in each group (meta**1**, layout**1** in Models**1**...)

When we say you can 😊, here's the URL:
tinyurl.com/Models2023T2

MetaEdit+ gives you nothing

- No effort to install
- No effort to upgrade language & co-evolve models
- No effort to work together on same models
- No effort on copying or locking before editing
- No effort on merging after modeling
- No effort to move between PCs or networks
- No effort to get VCS version history
- No effort to support multiple languages
- No effort to upgrade to new tool versions
- Just model.



MetaCase

Thank you!

Questions? Experiences? Arguments?

Co-evolution paper: Tue 3, 09:00 W1

Co-evolution demo: Thu 5, 16:15 B6